

VBA: MSAG-Style Output from Centerlines

Contributed by Bert Granberg
 15, Apr. 2008
 Last Updated 16, May. 2008

This sub procedure and three associated functions run in ArcMap against a selected set of polyline street centerline features whose attributes follow the data standards the Utah Transportation Data Model (UTDM).

The script produces an output text file similar to a Master Street Address Guide (MSAG) style. Please read description and disclaimer before using.

Report and questions or comments to bgranberg@utah.gov

```
Public Sub MSAGStyleOutputFromUTDM()
```

```
'Date: REVISED 4/15/08 14:15 AGRC-BG
```

```
'Description: This script generates a text-based output file
' with information similar to the needs/style of a
' Master Street Address Guide (MSAG). It output is
' generated from a selection of street centerline feature
' data in Utah Transportation Data Model (UTDM) format.
' The script tracks min and max address range values in
' for streets in each address coordinate system quadrant
' and does not account for cases where there are gaps in house
' number assignments (where a street is not contiguous as
' it runs across town)
```

```
' DISCLAIMER: THIS IS JUST A SAMPLE, YOU WILL NEED TO
' MODIFY &/OR EXTEND THIS SCRIPT AND DO EXTENSIVE TESTING
' TO ENSURE THAT YOUR OUTPUT GENERATED MEETS YOUR NEEDS.
```

```
' USE AT OWN RISK!!
```

```
'Requirements: Centerline address ranges, names, & directions must conform
' to UTDM standards. Selected streets features must have
' valid address range data. Centerline feature class must
' be the first layer (layer 0) in the ArcMap table of
' contents in 'Display Mode'
```

```
'Set ESN constant for output file example
```

```
Dim esnStr As String
esnStr = "911ESN"
```

```
'Set location for outfile
```

```
Dim outFileLocation As String
outFileLocation = "C:\msag.txt"
```

```
'Get Reference To Current ArcMap Session
```

```
Dim pMxDoc As IMxDocument
Dim pMap As IMap
Set pMxDoc = ThisDocument
Set pMap = pMxDoc.FocusMap
```

```
'Get the first layer in the table of contents, layer 0
'the featureclass you are using must be referenced by the
'first layer in the map
```

```
Dim pFLayer As IFeatureLayer
Dim pFClass As IFeatureClass
Dim pTable As ITable
Set pFLayer = pMap.Layer(0)
```

```

Set pFClass = pFLayer.FeatureClass
Set pTable = pFClass ' QI

'Get the current selected set of street features
Dim pFSel As IFeatureSelection
Dim pSelSet As ISelectionSet
Set pFSel = pFLayer 'QI
Set pSelSet = pFSel.SelectionSet

'declare variables and get the index number corresponding
'to each fields position
Dim lFIIndex As Integer
Dim lTIIndex As Integer
Dim rFIIndex As Integer
Dim rTIIndex As Integer
Dim preIndex As Integer
Dim sufIndex As Integer
Dim sNameIndex As Integer
Dim sTypeIndex As Integer
Dim sufIndex As Integer
Dim jurisIndex As Integer

Dim esnIndex As Integer
Dim acsIndex As Integer
Dim outStr As String

lFIIndex = pFClass.FindField("L_F_ADD")
lTIIndex = pFClass.FindField("L_T_ADD")
rFIIndex = pFClass.FindField("R_F_ADD")
rTIIndex = pFClass.FindField("R_T_ADD")
preIndex = pFClass.FindField("PRE_DIR")
sNameIndex = pFClass.FindField("S_NAME")
sTypeIndex = pFClass.FindField("S_TYPE")
sufIndex = pFClass.FindField("SUF_DIR")
jurisIndex = pFClass.FindField("CITY")
esnIndex = pFClass.FindField("ESN")
acsIndex = pFClass.FindField("ACS_ALIAS")

'set up tablesort to return results of a multi-field sort
Dim pTableSort As ITableSort
Set pTableSort = New TableSort 'VBA Query Interface
'the fields used and their order or precedence for the sort
pTableSort.Fields = ("CITY, PRE_DIR, S_NAME, S_TYPE, SUF_DIR")
pTableSort.Ascending("CITY") = True
pTableSort.Ascending("PRE_DIR") = True
pTableSort.Ascending("S_NAME") = True
pTableSort.Ascending("S_TYPE") = True
pTableSort.Ascending("SUF_DIR") = True
pTableSort.CaseSensitive("CITY") = False
pTableSort.CaseSensitive("PRE_DIR") = False
pTableSort.CaseSensitive("S_NAME") = False
pTableSort.CaseSensitive("S_TYPE") = False
pTableSort.CaseSensitive("SUF_DIR") = False
'sort on only selected features
Set pTableSort.SelectionSet = pSelSet

Set pTableSort.QueryFilter = Nothing

pTableSort.Sort Nothing

'get a cursor to iterate thru the features returned by the tablesort
Dim pCursor As ICursor
Set pCursor = pTableSort.Rows

'declare variable that track changes & high/low values in
'address attributes as we iterate through the sorted features below

```

```

Dim lastJuris As String
Dim lastComboString As String
Dim highVal As Long
Dim lowVal As Long
Dim lastDirStr As String
Dim lastStreetStr As String
Dim lastEsn As String
Dim lastAcs As String
Dim lastSuf As String
Dim currComboString As String

'set preconditions before looping structure
Dim pRow As IRow
Set pRow = pCursor.NextRow
highVal = 0
lowVal = 99999
lastJuris = convNull2Str(pRow.Value(jurisIndex))
lastComboString = convNull2Str(pRow.Value(preIndex)) & " " & _
    convNull2Str(pRow.Value(sNameIndex)) & " " & _
    convNull2Str(pRow.Value(sTypeIndex)) & " " & _
    convNull2Str(pRow.Value(suflIndex))
lastComboString = removeExtraSpaces(lastComboString)
lastDirStr = convNull2Str(pRow.Value(preIndex))
lastAcs = convNull2Str(pRow.Value(acsIndex))
lastSuf = convNull2Str(pRow.Value(suflIndex))
outStr = ""

'open text file for output & add field headers
Open outFileLocation For Output As #1
Print #1, addSpaces("PRE", 4, False) & addSpaces("STREET", 30, False) & _
    addSpaces("SUF", 4, False) & addSpaces("ALIAS", 8, False) & _
    addSpaces("LOW", 5, True) & " " & addSpaces("HI", 5, False) & _
    addSpaces("COMMUNITY", 20, False) & addSpaces("ESN", 7, False) & _
    addSpaces("DATE", 6, False)

'Begin Looping Structure: For each feature in the sorted features....
Do Until pRow Is Nothing

    currComboString = convNull2Str(pRow.Value(preIndex)) & " " & _
        convNull2Str(pRow.Value(sNameIndex)) & " " & _
        convNull2Str(pRow.Value(sTypeIndex)) & " " & _
        convNull2Str(pRow.Value(suflIndex))
    currComboString = removeExtraSpaces(currComboString)

    'check to see if the current feature has a new street ComboString string or
    'city/jurisdiction string
    If lastJuris <> convNull2Str(pRow.Value(jurisIndex)) Or lastComboString <> currComboString Then
        'NEW MSAG ENTRY ENCOUNTERED
        'write out previous entry
        Print #1, outStr
        Debug.Print outStr
        highVal = 0
        lowVal = 99999
    End If

    'Evaluate Left_From Value to see if new high or low is present
    If Not IsNull(pRow.Value(IFIndex)) Then

        If pRow.Value(IFIndex) > highVal Then
            'new high
            highVal = pRow.Value(IFIndex)
        End If
        If pRow.Value(IFIndex) < lowVal Then
            'new low
        End If
    End If
End If

```

```

    lowVal = pRow.Value(lFIndex)
End If

End If

'Evaluate Left_To Value to see if new high or low is present
If Not IsNull(pRow.Value(lTIndex)) Then

    If pRow.Value(lTIndex) > highVal Then
        'new high
        highVal = pRow.Value(lTIndex)
    End If
    If pRow.Value(lTIndex) < lowVal Then
        'new low
        lowVal = pRow.Value(lTIndex)
    End If
End If

'Evaluate Right_From Value to see if new high or low is present
If Not IsNull(pRow.Value(rTIndex)) Then

    If pRow.Value(rTIndex) > highVal Then
        'new high
        highVal = pRow.Value(rTIndex)
    End If
    If pRow.Value(rTIndex) < lowVal Then
        'new low
        lowVal = pRow.Value(rTIndex)
    End If
End If

'Evaluate Right_To Value to see if new high or low is present
If Not IsNull(pRow.Value(rFIndex)) Then

    If pRow.Value(rFIndex) > highVal Then
        'new high
        highVal = pRow.Value(rFIndex)
    End If
    If pRow.Value(rFIndex) < lowVal Then
        'new low
        lowVal = pRow.Value(rFIndex)
    End If
End If

'Prepare for next iteration of loop
'Set the values for variables that reference info about the last feature encountered
'use Replace function to remove spaces from lastStreetStr when fields are not populated

lastStreetStr = Replace(convNull2Str(pRow.Value(sNameIndex)) & " " & _
    convNull2Str(pRow.Value(sTypeIndex)), " ", "")
lastDirStr = convNull2Str(pRow.Value(preIndex))
lastJuris = convNull2Str(pRow.Value(jurisIndex))
lastSuf = convNull2Str(pRow.Value(suflIndex))

lastComboString = convNull2Str(pRow.Value(preIndex)) & " " & _
    convNull2Str(pRow.Value(sNameIndex)) & " " & _
    convNull2Str(pRow.Value(sTypeIndex)) & " " & _
    convNull2Str(pRow.Value(suflIndex))
lastComboString = removeExtraSpaces(lastComboString)
lastAcs = convNull2Str(pRow.Value(acsIndex))
outStr = addSpaces(lastDirStr, 4, False) & addSpaces(lastStreetStr, 30, False) & _
    addSpaces(lastSuf, 4, False) & addSpaces(lastAcs, 8, False) & _
    addSpaces(CStr(lowVal), 5, True) & addSpaces(CStr(highVal), 5, False) & _
    addSpaces(lastJuris, 20, False) & addSpaces(esnStr, 6, False) & " " & _

```

```

addSpaces(Format(Now, "mmddyy"), 6, False)

Set pRow = pCursor.NextRow

Loop

'Print results for the last street as this was not output within looping structure
Print #1, outStr
Debug.Print outStr
'close file for output
Close #1

End Sub

Public Function addSpaces(inValStr As String, desiredLen As Integer, addToFront As Boolean) As String

Dim x As Long
addSpaces = inValStr

x = Len(inValStr)

Do Until x >= desiredLen
    If addToFront Then
        addSpaces = " " & addSpaces
    Else
        addSpaces = addSpaces & " "
    End If
    x = x + 1
Loop

End Function

Public Function removeExtraSpaces(myStr As String) As String

Do Until InStr(myStr, " ") = 0
    myStr = Replace(myStr, " ", " ")
Loop
removeExtraSpaces = myStr

End Function

Public Function convNull2Str(inVal As Variant) As String

If IsNull(inVal) Then
    convNull2Str = ""
Else
    convNull2Str = Trim(CStr(inVal))
End If

End Function

```